An OpenSimplex approach: Performance optimization techniques for Hadoop Map Reduce system by auto tuning Hadoop system configuration parameters.

Vinay S. Khedekar Tejas J. Ghalsasi Momtaz M. Afredi

Department of Computer Science
California State University, Fullerton
Fullerton, California

***Abstract*— The objective of this examination venture is to direct extensive research on Hadoop MapReduce framework's design parameters, to grow new execution improvement strategies for Hadoop MapReduce framework. As accomplishing ideal outcomes from a Hadoop usage starts with exceedingly proficient framework designs and default estimations of these parameters don't bring about agreeable execution and in this way, it is vital to tune them. The exploration work primarily concentrates on the profound investigation of 190 setup parameters of Hadoop framework which incorporates investigation of parameters of Linux document framework, parameters of Hadoop, Guide/Diminish Particular Arrangements and Hadoop work parameter. Likewise, the objective of research ponder is to build up a without dimensionality strategy which can naturally tune the setup parameters by considering the cross-parameter conditions. The new 'OpenSimplex approach' an advancement procedure which will break down the Hadoop framework cross-parameter arrangements given use setup and tune them to accomplish wanted framework execution.**

Keywords—Hadoop; MapReduce; configuration parameter; auto-tuning; parameter optmization

I. INTRODUCTION

Numerous associations are persistently gathering monstrous measures of datasets from different sources such as the World Wide Web, sensor systems, and informal organizations. The capacity to perform adaptable and opportune investigation on these unstructured datasets is a high need for some ventures. It has moved toward becoming troublesome for conventional database frameworks to process these ceaselessly developing datasets. Hadoop MapReduce has turned into a noteworthy processing innovation in help of huge information examination [1, 2]. Hadoop has gotten a wide take-up from the group due to its noteworthy highlights, for example, high versatility, adaptation to non-critical failure, and information parallelization. It consequently disseminates information and parallelizes calculation over a bunch of PC hubs [3– 7].

Regardless of these noteworthy highlights, Hadoop is a vast and complex structure, which has a number of segments that interface with each different over various PC hubs. The execution of a Hadoop work is touchy to every segment of the Hadoop structure including the basic equipment, arrange foundation, and Hadoop setup parameters, which are more than 190. Later inquiries about demonstrate that the parameter settings of the Hadoop system assume a basic part in the execution of Hadoop. A little change in the arrangement parameter settings can have a noteworthy effect on the execution of a Hadoop work [8]. Physically tuning the ideal or close ideal estimations of these parameters is a testing errand and furthermore a tedious procedure. In expansion, the Hadoop system has a discovery like component, which makes it to a great degree hard to locate a numerical model or a goal work, which speaks to a connection among the parameters. The substantial parameter space together with the mind-boggling connections among the design parameters additionally builds the many-sided quality of a manual tuning process. Along these lines, a compelling and programmed way to deal with tuning Hadoop parameters has turned into a need.

II. EXISTING CONFIGURATION TUNING APPROACHES

Various research works have been proposed to consequently tune Hadoop parameter settings. The general guideline (ROT) proposed by mechanical experts [9– 11] is only a typical practice to tune Hadoop parameter settings. The Starfish streamlining agent [12, 13] advances the execution of a Hadoop work in view of the activity profile and a cost demonstrate [14]. The activity profile is gathered at a fine granularity with nitty gritty data. Nonetheless, gathering the itemized execution profile of an occupation acquires a high overhead, which overestimates the qualities for some arrangement parameters. Besides, the Starfish streamlining agent isolates the hunt space into subspaces in the advancement

procedure, which overlooks the connections among the design parameters. PPABS [15] naturally tunes Hadoop parameter settings in view of the executed activity profiles. Profiling and Performance Analysis-based Framework (PPABS) utilizes K-means++ to group the employments into comparable classes. It applies reenacted toughening to look for ideal parameter esteems and executes an example acknowledgment strategy to decide the class that another activity has a place with. Be that as it may, PPABS can't tune the parameter settings for another activity, which does not have a place with any of the pre-characterized classes.

Gunther, an inquiry based framework proposed in [16] consequently scans for ideal parameter values for the arrangement parameters utilizing hereditary calculation. One basic restriction of Gunther is that it doesn't have a wellness work in the actualized hereditary calculation. Gunther assesses the wellness of an arrangement of parameter esteems by running a Hadoop work physically, which is a tedious process. Panacea [17] improves Hadoop applications in light of a procedure of tuning the design parameter settings. Like Starfish, Panacea additionally partitions the inquiry space into subspaces and after that scans for ideal esteems inside pre-characterized ranges. The work exhibited in [18] proposes an execution assessment show, which concentrates on the effect of the Hadoop setup settings from the parts of equipment, programming, and system. Tuning the design parameters of Hadoop requires the learning of the inner progression of the Hadoop system and the between conditions among its setup parameters. This is on the grounds that the estimation of one parameter can significantly affect alternate parameters. It ought to be pointed out that none of the previously mentioned works considers the between conditions among Hadoop arrangement parameters.

In this paper, we upgrade the execution of Hadoop via naturally tuning its arrangement parameter settings. Based on the running records of Hadoop jobs, which can be either CPU intensive or IO intensive, we utilize quality articulation programming (GEP) system to assemble a target function, which speaks to a correlation among the Hadoop setup parameters. To the best of our insight, this is the main work that numerically portrays the between conditions among the Hadoop arrangement parameters when tuning the execution of Hadoop. For the motivation behind setup parameter improvement, molecule swarm enhancement (PSO) [19, 20] is utilized that makes utilization of the GEP built target capacity to scan for a set of ideal or close ideal estimations of the setup parameters. Not at all like other enhancement works that partition the inquiry space into subspaces, the executed PSO considers the entirety look space in the advancement procedure keeping in mind the end goal to keep up the between conditions among the setup parameters.

To assess the execution of the proposed work, we run two runs of the mill Hadoop MapReduce applications, that is, Word Count and Sort, which are CPU and Input/Output (I/O) concentrated, individually. The execution of the proposed work is at first assessed on an exploratory Hadoop group arranged with eight Virtual Machines (VMs) and accordingly on another Hadoop group arranged with 16 VMs. The trial comes about demonstrate that the proposed work improves the execution of Hadoop by and large 67% on the Word Count application and 46% on the Sort application, individually, contrasted and its default settings. The proposed work likewise beats both ROT and the Starfish show in Hadoop execution streamlining.

III. HADOOP CORE PARAMETERS

The Hadoop structure has more than 190 tunable arrangement parameters that enable clients to deal with the stream of a Hadoop work in various stages amid the execution procedure. Some of them are center parameters and significantly affect the execution of a Hadoop work [12,16]. The center parameters are quickly displayed in Table I.

2.1. io.Sort.Factor

This parameter decides the quantity of records (streams) to be converged amid the arranging procedure of guide undertakings. The default esteem is 10, yet expanding its esteem enhances the use of the physical memory what's more, diminishes the overhead in IO operations.

2.2. io.Sort.mb

Amid an occupation execution, the yield of a guide errand isn't specifically built into the hard circle yet is composed into an in-memory cradle, which is doled out to each guide undertaking. The measure of the in-memory cradle is determined through the io.sort.mb parameter. The default estimation of this parameter is 100MB. The suggested an incentive for this parameter is in the vicinity of 30% and 40% of the Java_Opts esteem and ought to be bigger than the yield size of a guide assignment, which limits the number of spill records [11].

Table I. Hadoop core configuration parameters.

Configuration Parameters	Default Values
io.sort.factor	10
io.sort.mb	100
io.sort.spill.percent	0.8
mapred.reduce.tasks	1
mapreduce.tasktracker.map. tasks.maximum	2
mapreduce.tasktracker.reduce. tasks.maximum	2
mapred.child.java.opts	200
mapreduce.reduce.shuffle. input.buffer.percent	0.70
mapred.reduce.parallel.copies	5
mapred.compress.map.output	False
mapred.output.compress	False

2.3. io.Sort.Spill.Percent

Default estimation of this parameter is 0.8 (80%). At the point The when an in-memory cushion is topped off to 80%, the

information of the in-memory cradle (io.sort.mb) ought to be spilled into the hard circle. It is suggested that the estimation of io.sort.spill.percent ought not be under 0.50.

2.4. Mapred.Reduce.Tasks

This parameter can significantly affect the execution of a Hadoop work [21]. The default esteem is 1. The ideal estimation of this parameter is predominantly subject to the measure of an info dataset and the quantity of diminish openings arranged in a Hadoop group. Setting few lessen assignments for work diminishes the overhead in setting up undertakings on a little info dataset, while setting a substantial number of decrease errands enhances the hard plate IO usage on a huge information dataset. The prescribed number of lessen undertakings is 90% of the aggregate number of decrease spaces arranged in a group [8].

2.5. Mapreduce.Tasktracker.map.Tasks.Maximum, mapreduce.Tasktracker.Reduce.Tasks.Maximum

These parameters characterize the quantity of the guide and diminish assignments that can be executed at the same time on each group hub. Expanding the estimations of these parameters builds the usage of CPUs and physical memory of the bunch hub, which can enhance the execution of a Hadoop work. The ideal estimations of these parameters are subject to the quantity of CPUs, the quantity of centers in every CPU, multi-threading ability, and the computational multifaceted nature of an occupation. The prescribed esteems for these parameters are the quantity of CPU centers short 1 as long as the group hub has adequate physical memory [9– 11]. One CPU is saved for different administrations in Hadoop, for example, Data Node and Task Tracker.

2.6. Mapred.Child.Java.Opts

This is a memory related parameter and the primary possibility for JVM tuning. The default esteem is – Xmx200m, which gives at most 200MB physical memory to every tyke undertaking. Expanding the estimation of Java_Opt lessens spill operations to yield delineate into the hard plate, which can enhance the execution of an occupation. Naturally, each work hub uses 2.8GB physical memory [11]. The specialist hub allocates 400MB to the guide stage (i.e., two guide openings), 400MB to the decrease stage (i.e., two diminish spaces) and 1000MB to each Data Node and Task Tracker that keep running on the laborer hub.

2.7. Mapred.Compress.map.Output, mapred.Output.Compress

These two parameters are identified with the hard plate IO and system information exchange operations. Boolean esteems are utilized to decide if the guide yield and the lessen yield should be compacted. Empowering the pressure of the guide and lessen yields for an occupation can accelerate the hard plate IO and limit the overhead in information rearranging over the system.

IV. How Configuration Parameters Plays Critical Role in Hadoop

In this section, we will be exploring configuration parameters that help in tuning in Hadoop. Some configuration we will be looking into are JVM, OS, and BIOS configuration that will show the configurations and how it helps make it a critical role in Hadoop.

A. JVM Configuration

In this segment, we talk about various JVM charge line switches and their potential effect on execution of Hadoop workloads.

On 64-bit Oracle JDK6 refresh 25 JVM, compacted pointers are empowered as a matter of course. On the off chance that you are utilizing a more established rendition of JDK and compacted pointers are crippled, explore different avenues regarding empowering them. Packed pointers decrease memory impression. We saw over 3% change in execution by empowering packed pointers on Cluster A. One-sided securing highlight Oracle Hot Spot JDK enhances execution in circumstances with un-fought locks. Given the engineering of Hadoop structure, one-sided bolting ought to for the most part enhance execution. On Cluster A, we saw over 5% change by empowering one-sided bolting. Prophet JVM advancements empowered by order line banners, for example, Aggressive Opts, Use Compressed Strings, and Use String Cache can affect Hadoop execution. Have a go at exploring different avenues regarding these banners. We, in any case, saw 2% debasement in execution by empowering Aggressive Opts signal on Cluster A. Check whether the JVM is coming up short on code store. Increment code reserve estimate if essential. Try different things with Use NUMA and Use Large Pages JVM banners. Perform point by point GC log examination and tuning of the guide and decrease JVM forms. We saw a 3% change in execution by tuning GC signals on Cluster A.

B. OS Configuration

This area presents data about effect of tuning Linux OS properties on Hadoop execution. Certain Linux conveyances bolster EXT4 as the default document framework sort. On the off chance that you are utilizing another sort of record framework, try different things with EXT4 document framework. We saw 9% execution changes by utilizing EXT4 record framework over EXT3 on Cluster A.

Naturally, every document read operation triggers a circle compose operation for keeping up last access time of the record. Impair this logging utilizing noatime, nodirtaime FS characteristics. Explore different avenues regarding different FS tuning traits, for example, degree, flex_bg, boundary and so forth. On Cluster B, we saw 15% change in execution by utilizing noatime FS trait. Linux parts bolster 4 distinct sorts of I/O schedulers – CFQ, due date, no-operation, and expectant. Explore different avenues regarding distinctive decisions of I/O scheduler, particularly CFQ and due date. On Cluster B, CFQ scheduler performed

15% superior to due date scheduler. Linux OS points of confinement, for example, max open record descriptors and epoll breaking points can affect execution, try different things with these cutoff points. We, in any case, saw relapse of 1% by expanding open fd point of confinement to 16K from its default estimation of 1K on Cluster A.

C. BIOS Configuration

In this area, we talk about a portion of the BIOS parameters that could conceivably affect Hadoop execution.

Local charge lining (NCQ) highlight of current hard drives enhances I/O execution by streamlining drive head development. Try different things with AHCI alternative in BIOS, which can be utilized to empower NCQ mode. At the point when all the CPU centers on the equipment are not completely used, the processor could downsize CPU recurrence and different assets, for example, Hyper Transport joins. Try different things with ACPI and other power-related BIOS choices. We saw 1% execution change by debilitating force sparing mode in the BIOS. This perception was made on Cluster A. On some AMD processor-based frameworks, North Bridge recurrence and width are powerfully tuned to diminish control utilization. On the off chance that memory transmission capacity is a bottleneck, try different things with choices that can be utilized to adjust North Bridge recurrence and width settings. On Cluster A, we saw 2% change in execution by tuning North Bridge settings. Present day AMD processors bolster an element called HT helps (a.k.a. test channels). This element lessens activity on memory interconnects to the detriment of some segment of L3 reserve. Try different things with HT helps settings; incapacitate it if your activity is touchy to L3 reserve measure.

V. WHAT IS AN OPEN-SIMPLEX APPROACH

The OpenSimplex approach is, auto-tuning the default Hadoop configuration parameters by finding the correlation and cross communication dependencies among configuration parameters and optimizing accordingly.

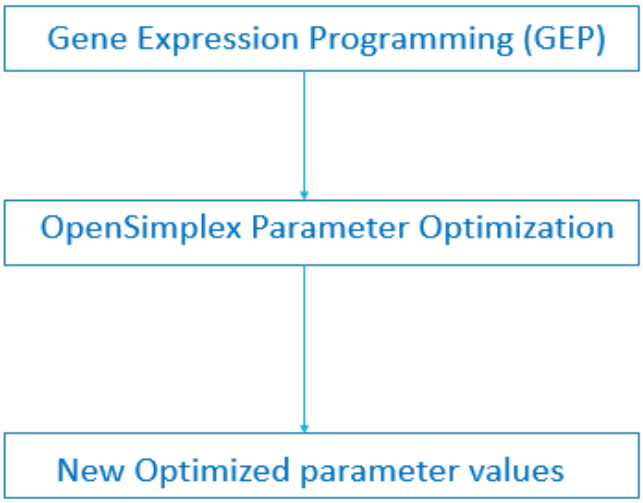

Fig. 1 Steps of OpenSimplex approach.

OpenSimplex is a two-step process-
Step 1: Find the correlation among Hadoop configuration parameters
Step 2: Optimize the configuration parameters

The OpenSimplex approach involves two stages.

A. Gene expression programming (GEP) technique

To build an objective function which represents a correlation among the Hadoop configuration parameters. This is the first work that mathematically describes the inter-dependencies among the Hadoop configuration parameters

B. OpenSimlex Parameter optimization (OSP)

For the purpose of configuration parameter optimization, we make use of the GEP constructed objective function to search for a set of optimal or near optimal values of the configuration parameters.

The implemented PSO considers the whole search space in the optimization process in order to maintain the inter-dependencies among the configuration parameters.

CONCLUSION

In this section, it talked about a portion of the accepted procedures in tuning diverse parts of the Hadoop structure. Arrangement tuning of the considerable number of segments of Hadoop stack is an essential exercise and can offer a colossal execution result. Distinctive Hadoop workloads will have diverse attributes, so it is essential to explore different avenues regarding diverse tuning choices. Examples shown were through JVM, OS, and BIOS configurations.

VI. IMPLEMENTATION OF OPENSIMPLEX APPROACH

(Future Work: Will be completed in final submission)

References

[1] Dean J, Ghemawat S. MapReduce: Simplified data processing on large clusters. In Proceedings of the 6th Conference on Symposium on Opearting Systems Design & Implementation - Volume 6, 2004; 10.

[2] Apache Hadoop. Apache. [Online]. Available: http://hadoop.apache.org/. [Accessed: 18-Feb-2015].

[3] Khan M, Ashton PM, Li M, Taylor GA, Pisica I, Liu J. Parallel detrended fluctuation analysis for fast event detection

[4] on massive PMU data. Smart Grid, IEEE Transactions 2015; 6(1):360–368.

[5] Khan M, Li M, Ashton P, Taylor G, Liu J. Big data analytics on PMU measurements. In Fuzzy Systems and

[6] Knowledge Discovery (FSKD), 2014 11th International Conference on, 2014; 715–719.

[7] Kang U, Tsourakakis CE, Faloutsos C. PEGASUS: Mining peta-scale graphs. Knowledge and Information Systems

[8] 2011; 27(2):303–325.

[9] Panda B, Herbach JS, Basu S, Bayardo RJ. PLANET: Massively parallel learning of tree ensembles with

[10] MapReduce. Proceedings of the VLDB Endowment 2009; 2(2):1426–1437.

[11] Pavlo A, Paulson E, Rasin A. A comparison of approaches to large-scale data analysis. In SIGMOD '09 Proceedings

[12] of the 2009 ACM SIGMOD International Conference on Management of Data, 2009; 165–178.

[13] Babu S. Towards automatic optimization of MapReduce programs. Proc. 1st ACM Symp. Cloud Comput. - SoCC

[14] '10, 2010; 137.

[15] Hadoop Performance Tuning. [Online]. Available: https://hadoop-toolkit.googlecode.com/files/White paper-

[16] HadoopPerformanceTuning.pdf. [Accessed: 23-Feb-2015].

[17] 7 tips for Improving MapReduce Performance. 2009. [Online]. Available: http://blog.cloudera.com/blog/2009/12/

[18] 7-tips-for-improving-mapreduce-performance/. [Accessed: 20-Feb-2015].

[19] White T. Hadoop:The Definitive Guide (3rd edn). Yahoo press: Sebastopol, CA, USA, 2012; 688.

[20] Herodotou H, Babu S. Profiling, what-if analysis, and cost-based optimization of MapReduce programs. PVLDB

[21] 2011; 4(11):1111–1122.

[22] Herodotou H, Lim H, Luo G, Borisov N, Dong L, Cetin FB, Babu S. Starfish: a self-tuning system for big data

[23] analytics. In CIDR, 2011, 261–272.

[24] Herodotou H. Hadoop Performance Models. 2011. [Online]. Available: http://www.cs.duke.edu/starfish/files/

[25] hadoop-models.pdf. [Accessed: 22-Oct-2013].

[26] Wu D, Gokhale AS. A self-tuning system based on application profiling and performance analysis for optimizing

[27] Hadoop MapReduce cluster configuration. In 20th Annual International Conference on High Performance Computing,

[28] HiPC 2013, Bengaluru (Bangalore), Karnataka, India, December 18–21, 2013, 2013; 89–98.

[29] Liao G, Datta K, Willke TL. Gunther: search-based auto-tuning of Mapreduce. In Proceedings of the 19th International

[30] Conference on Parallel Processing, 2013; 406–419.

[31] Liu J, Ravi N, Chakradhar S, Kandemir M. Panacea: towards holistic optimization of MapReduce applications. In

[32] Proceedings of the Tenth International Symposium on Code Generation and Optimization, 2012; 33–43.

[33] Li Y, Wang K, Guo Q, Li X, Zhang X, Chen G, Liu T, Li J. Breaking the boundary for whole-system performance

[34] optimization of big data. In Low Power Electronics and Design (ISLPED), 2013 IEEE International Symposium on,

[35] 2013; 126–131.

[36] Kennedy J, Eberhart R. Particle swarm optimization. In Proceedings., IEEE International Conference on Neural

[37] Networks 1995. 1995; 4:1942–1948.

[38] Eberhart R, Kennedy J. A new optimizer using particle swarm theory. In Micro Machine and Human Science, 1995.

[39] MHS '95., Proceedings of the Sixth International Symposium on, 1995; 39–43.

[40] Khan M, Jin Y, Li M, Xiang Y, Jiang C. Hadoop performance modeling for job estimation and resource provisioning.